**THAT WHICH IS CLAIMED IS:**

1. A method of searching a database, the method comprising:

generating a hash key value based on a plurality of selector values;

selecting an entry in the database having an address corresponding to the hash key value, wherein entries in the database include corresponding hash values;

evaluating the selected entry to determine if the entry in the database corresponds to the plurality of selector values;

incrementing the address corresponding to the hash key value if the selected entry does not correspond to the plurality of selector values;

wherein the selecting, the evaluating and the incrementing are repeated until the hash value included in selected entry has a value which indicates that entries subsequent to the selected entry will not correspond to the plurality of selector values.

2. A method according to Claim 1, wherein the selecting, the evaluating and the incrementing are repeated until an entry corresponding to the plurality of selector values is reached or until the hash value included in the selected entry has a value which indicates that entries subsequent to the selected entry will not correspond to the plurality of selector values.

3. A method according to Claim 1, wherein the selecting, the evaluating and the incrementing are repeated until the selected entry is a null entry.

4. A method according to Claim 1, wherein the selecting, the evaluating and the incrementing are repeated until the selected entry has a hash value greater than the hash key value.

5. The method of Claim 2, further comprising:

providing the selected entry if the selected entry corresponds to the plurality of selector values; and

providing an indicator of failure of the search if the selected entry includes a hash value other than the hash key value or the selected entry has a null value.

6. The method of Claim 1, wherein generating a hash key value based on a plurality of selector values comprises encrypting the selector values to provide the hash key value.

7. The method of Claim 6, wherein encrypting the selector values to provide the hash key value comprises:

grouping the plurality of selector values into blocks having a predefined number of bits;

padding the blocks of grouped selector values to the predefined number of bits;

encrypting the padded blocks; and

truncating the encrypted padded blocks to a number of bits in the hash key value to provide the hash key value.

8. The method of Claim 7, wherein encrypting the padded blocks comprises encrypting the padded blocks using Cipher-Block-Chaining encryption mode of Data Encryption Standard (DES-CBC) encryption.

9. The method of Claim 8, wherein the database comprises an Internet Protocol Security (IPSec) security association database, the plurality of selector values comprise IPSec selector fields and the predefined number of bits comprises 64 bits.

10. The method of Claim 1, wherein the database comprises an Internet Protocol Security (IPSec) security association database and the plurality of selector values comprise IPSec selector fields.

11. The method of Claim 10, wherein the database has a size of about four times a maximum number of supported security associations.

12. The method of Claim 1, wherein the database is contained in a circular memory and wherein incrementing the address comprises:

incrementing the address to a next consecutive address if the address is less than a maximum address of the circular memory; and

setting the address to a first address of the circular memory if the address is equal to the maximum address of the circular memory.

13. The method of Claim 12, wherein the selecting, the evaluating and the incrementing are repeated until a hash value of the selected entry is less than a hash value of a previous selected entry and the hash value of the selected entry is greater than the hash key value.

14. A method of inserting data for entries into a database, comprising:

generating a hash key value based on a plurality of selector values associated with the data for entry into the database; and

incorporating the data and the hash key value as an entry into the database at an address in the database which maintains entries in the database in hash key value sequence such that a linear search for the data from an address corresponding to the hash key value will result in the data being located by examining entries in consecutive addresses in the database before an address in the database without an entry is reached.

15. The method of Claim 14, wherein incorporating the data and the hash key value as an entry into the database is carried out utilizing only atomic read and/or write operations such that inserting data for entries into the database can be carried out simultaneously with a search of the database.

16. The method of Claim 14, wherein incorporating the data and the hash key value as an entry into the database comprises:

determining an address in the database closest to an address in the database corresponding to the hash key value for which the database does not have an entry;

inserting the data and the hash key value as an entry in the database at the determined address if the determined address is the address corresponding to the hash key value;

inserting the data and the hash key value in the database at a next subsequent address after the address corresponding to the hash key value which is after an address of an entry in the database having an associated hash value of less than or equal to the hash key value and before an entry in the database having an associated hash value of

greater than the hash key value if the entry located at the address corresponding to the hash key value is not empty; and

shifting data and hash key values from the next subsequent address to an address just prior to the determined address to provide entries in the database from an address just after the next subsequent address to the determined address if the entry located at the address corresponding to the hash key value is not empty.

17. The method of Claim 16, wherein the database comprises a circular memory, the method further comprising inserting the data and the hash key value at a second next subsequent address after the address corresponding to the hash key value, where the second next subsequent address is immediately after an address of an entry in the database having an associated value of less than a hash value of an entry in the database at the second next subsequent address and either the hash key value is greater than the second next subsequent address or the hash key value is both less than the second next subsequent address and less than the hash value of the entry in the database at the second next subsequent address.

18. The method of Claim 14, wherein generating a hash key value based on a plurality of selector values comprises encrypting the selector values to provide the hash key value.

19. The method of Claim 18, wherein encrypting the selector values to provide the hash key value comprises:

grouping the plurality of selector values into blocks having a predefined number of bits;

padding the blocks of grouped selector values to the predefined number of bits;

encrypting the padded blocks; and

truncating the encrypted padded blocks to a number of bits in the hash key value to provide the hash key value.

20. The method of Claim 19, wherein encrypting the padded blocks comprises encrypting the padded blocks using Cipher-Block-Chaining encryption mode of Data Encryption Standard (DES-CBC) encryption.

21. The method of Claim 19, wherein the database comprises an Internet Protocol Security (IPSec) security association database, the plurality of selector values comprise IPSec selector fields and the predefined number of bits comprises 64 bits.

22. The method of Claim 14, wherein the database comprises an Internet Protocol Security (IPSec) security association database and the plurality of selector values comprise IPSec selector fields.

23. The method of Claim 22, wherein the database has a size of about four times a maximum number of supported security associations.

24. A method of deleting data from a database, the method comprising:

generating a hash key value based on a plurality of selector values associated with the data for deletion from the database;

locating an entry in the database which includes the data and the hash key value;

deleting the located entry; and

reordering a subset of the entries in the database so as to maintain entries in the database in hash key value sequence such that a linear search for the data from an address corresponding to the hash key value will result in the data being located by examining entries in consecutive addresses in the database before an address in the database without an entry is reached.

25. The method of Claim 24, wherein deleting the located entry and reordering a subset of the entries in the database are carried out utilizing only atomic read and/or write operations such that deleting data from the database can be carried out simultaneously with a search of the database.

26. The method of Claim 24, wherein locating an entry in the database comprises:

selecting an entry in the database having an address corresponding to the hash key value, wherein entries in the database include corresponding hash values;

evaluating the selected entry to determine if the entry in the database corresponds to the plurality of selector values;

incrementing the address corresponding to the hash key value if the selected entry does not correspond to the plurality of selector values;

wherein the selecting, the evaluating and the incrementing are repeated until an entry corresponding to the plurality of selector values is reached.

27. The method of Claim 24, wherein deleting the located entry and reordering entries in the database comprises replacing the located entry in the database with a null entry if a next subsequent entry after the located entry is a null entry.

28. The method of Claim 27, wherein deleting the located entry and reordering entries in the database further comprises replacing the located entry in the database with a null entry if the next subsequent entry after the located entry is at an address in the database corresponding to a hash value of the next subsequent entry after the located entry.

29. The method of Claim 28, wherein deleting the located entry and reordering entries in the database further comprises replacing an entry at a current address of the database with an entry at a next subsequent address in the database if the current address is not before an address of the located entry and the next subsequent entry is not at an address in the database corresponding to a hash value of the next subsequent entry after the located entry.

30. The method of Claim 25, wherein deleting the located entry and reordering entries in the database further comprises replacing an entry at a current address of the database with an entry at a next subsequent address in the database if the current address is not before an address of the located entry and the next subsequent entry not at an address in the database corresponding to a hash value of the next subsequent entry after the located entry or if the next subsequent entry is a null entry.

31. The method of Claim 24, wherein generating a hash key value based on a plurality of selector values comprises encrypting the selector values to provide the hash key value.

32. The method of Claim 31, wherein encrypting the selector values to provide the hash key value comprises:

grouping the plurality of selector values into blocks having a predefined number of bits;

padding the blocks of grouped selector values to the predefined number of bits;

encrypting the padded blocks; and

truncating the encrypted padded blocks to a number of bits in the hash key value to provide the hash key value.

33. The method of Claim 32, wherein encrypting the padded blocks comprises encrypting the padded blocks using Cipher-Block-Chaining encryption mode of Data Encryption Standard (DES-CBC) encryption.

34. The method of Claim 33, wherein the database comprises an Internet Protocol Security (IPSec) security association database, the plurality of selector values comprise IPSec selector fields and the predefined number of bits comprises 64 bits.

35. The method of Claim 24, wherein the database comprises an Internet Protocol Security (IPSec) security association database and the plurality of selector values comprise IPSec selector fields.

36. The method of Claim 35, wherein the database has a size of about four times a maximum number of supported security associations.

37. A system searching a database, comprising:

means for generating a hash key value based on a plurality of selector values;

means for selecting an entry in the database having an address corresponding to the hash key value, wherein entries in the database include corresponding hash values;

means for evaluating the selected entry to determine if the entry in the database corresponds to the plurality of selector values;

means for incrementing the address corresponding to the hash key value if the selected entry does not correspond to the plurality of selector values;

means for repeatedly selecting, evaluating and incrementing until the selected entry has a null value or the hash value included in selected entry has a value other than the hash key value.

38. A system for inserting data for entries into a database, comprising:

means for generating a hash key value based on a plurality of selector values associated with the data for entry into the database; and

means for incorporating the data and the hash key value as an entry into the database at an address in the database which maintains entries in the database in hash key value sequence such that a linear search for the data from an address corresponding to the hash key value will result in the data being located by examining entries in consecutive addresses in the database before an address in the database without an entry is reached.

39. A system deleting data from a database, comprising:

means for generating a hash key value based on a plurality of selector values associated with the data for deletion from the database;

means for locating an entry in the database which includes the data and the hash key value;

means for deleting the located entry; and

means for reordering a subset of the entries in the database so as to maintain entries in the database in hash key value sequence such that a linear search for the data from an address corresponding to the hash key value will result in the data being located by examining entries in consecutive addresses in the database before an address in the database without an entry is reached.

40. A computer program product for searching a database, comprising:

a computer-readable storage medium having computer-readable program code embodied therein, the computer readable program code comprising:

computer-readable program code which generates a hash key value based on a plurality of selector values;

computer-readable program code which selects an entry in the database having an address corresponding to the hash key value, wherein entries in the database include corresponding hash values;

computer-readable program code which evaluates the selected entry to determine if the entry in the database corresponds to the plurality of selector values;

computer-readable program code which increments the address corresponding to the hash key value if the selected entry does not correspond to the plurality of selector values;

computer-readable program code which repeatedly selects, evaluates and increments until the selected entry has a null value or the hash value included in selected entry has a value other than the hash key value.

41. A computer program product for inserting data for entries into a database, comprising:

a computer-readable storage medium having computer-readable program code embodied therein, the computer readable program code comprising:

computer-readable program code which generates a hash key value based on a plurality of selector values associated with the data for entry into the database; and

computer-readable program code which incorporates the data and the hash key value as an entry into the database at an address in the database which maintains entries in the database in hash key value sequence such that a linear search for the data from an address corresponding to the hash key value will result in the data being located by examining entries in consecutive addresses in the database before an address in the database without an entry is reached.

42. A computer program product for deleting data from a database, comprising:

a computer-readable storage medium having computer-readable program code embodied therein, the computer readable program code comprising:

computer-readable program code which generates a hash key value based on a plurality of selector values associated with the data for deletion from the database;

computer-readable program code which locates an entry in the database which includes the data and the hash key value;

computer-readable program code which deletes the located entry; and

computer-readable program code which reorders a subset of the entries in the database so as to maintain entries in the database in hash key value sequence such that a linear search for the data from an address corresponding to the hash key value will result in the data being located by examining entries in consecutive addresses in the database before an address in the database without an entry is reached.

43. A data structure comprising:

a plurality of data entries, each of the plurality of data entries including a hash value associated with the data and which is generated from a plurality of selector values which uniquely identify the data and having an address associated therewith;

a plurality of null entries having an associated address other than an address in the data structure associated with a data entry;

wherein the address associated with a data entry is based on the hash value of the data entry such that a linear search for the data entry from an address corresponding to the hash value of the data entry will result in the data entry being located by examining entries in consecutive addresses before an address with a null entry is reached.

44. The data structure of Claim 43, wherein the addresses associated with the data entries are in ascending order based on the hash values of the data entries.

45. The data structure of Claim 43, wherein the addresses associated with the data entries are in descending order based on the hash values of the data entries.

46. The data structure of Claim 43, wherein the addresses are consecutive addresses.

47. The data structure of Claim 46, wherein a next consecutive address from a last address of the data structure is a first address of the data structure.

48. The data structure of Claim 43, wherein a total number of data entries and null entries in the data structure is greater than a total number of potential unique data entries such the a total number of addresses in the data structure is greater than the total number of potential unique entries.

49. The data structure of Claim 48, wherein the total number of addresses is about four times the total number of potential unique entries.

50. The data structure of Claim 43, wherein the data structure comprises an Internet Protocol Security (IPSec) Security Association Database (SAD), the data of the data entries comprises IPSec security association (SA) information and the hash values comprise hash keys generated from selector fields of the SAs.

51. A system for managing Internet Protocol Security (IPSec) security associations (SAs), comprising:

a hash key generator configured to generate hash key values based on modified selectors fields of Internet Protocol (IP) packets, the modified selector fields identifying a SA associated with the packet; and

a SA data structure operably associated with the hash key generator and configured to store SA information and associated hash key values in hash-ordered sequence such that a linear search for a SA from an address of the data structure corresponding to a hash key value generated from the modified selector fields identifying the SA will result in the SA being located by examining SAs at consecutive addresses before an address with a null entry is reached.

52. A system according to Claim 51, wherein the SA data structure is further configured to incorporate SAs and their corresponding hash key values into the data structure at an address in the data structure which maintains the SAs in the data structure in hash key value sequence such that a linear search for a SA from an address of the data structure corresponding to a hash key value generated from the modified selector fields identifying the SA will result in the SA being located by examining SAs at consecutive addresses before an address with a null entry is reached.

53. A system according to Claim 51, wherein the SA data structure is further configured to locate a SA in the database for deletion, delete the located SA and reorder SAs in the data structure so as to maintain the SAs in the data structure in hash key value sequence such that a linear search for a SA from an address of the data structure corresponding to a hash key value generated from the modified selector fields identifying the SA will result in the SA being located by examining SAs at consecutive addresses before an address with a null entry is reached.

54. A method of searching a database stored in a circular memory, the method comprising:

generating a hash key value based on a plurality of selector values;

selecting an entry in the database having an address corresponding to the hash key value, wherein entries in the database include corresponding hash values;

evaluating the selected entry to determine if the entry in the database corresponds to the plurality of selector values;

evaluating most significant bits of a hash value of the selected entry and most significant bits of the hash key value to determine if a wrap condition has occurred;

inverting the most significant bits of the hash value of the selected entry and the most significant bits of the hash key value if a wrap condition has occurred;

comparing the hash key value to the hash value of the selected entry to determine if the hash value of the selected entry is greater than the hash key value; and

incrementing the address corresponding to the hash key value if the selected entry does not correspond to the plurality of selector values and the hash value of the selected entry is greater than the hash key value.

55. The method of Claim 54, wherein the database comprises an Internet Protocol Security (IPSec) security association database and the plurality of selector values comprise IPSec selector fields.

56. The method of Claim 54, wherein the database has a size of about four times a maximum number of supported security associations, the most significant bits comprises the two most significant bits and evaluating most significant bits comprises

determining if the two most significant bits of the hash value of the current entry are "11" and the two most significant bits of the hash key value are "00" or if the two most significant bits of the hash value of the selected entry are "00" and the two most significant bits of the hash key value are "11".

57. The method of Claim 54, wherein incrementing the address comprises:

incrementing the address to a next consecutive address if the address is less than a maximum address of the circular memory; and

setting the address to a first address of the circular memory if the address is equal to the maximum address of the circular memory.

58. A method of inserting data for entries into a database stored in a circular memory, comprising:

generating a hash key value based on a plurality of selector values associated with the data for entry into the database;

selecting an entry in the database having an address corresponding to the hash key value, wherein entries in the database include corresponding hash values;

determining an end of a cluster of database entries by incrementing the address corresponding to the hash key value and selecting the corresponding entry in the database until an entry after the selected entry is empty;

evaluating most significant bits of a hash value of the selected entry and most significant bits of the hash key value to determine if a wrap condition has occurred;

inverting the most significant bits of the hash value of the selected entry and the most significant bits of the hash key value if a wrap condition has occurred;

comparing the hash key value to the hash value of the selected entry to determine if the hash value of the selected entry is greater than the hash key value;

copying the selected entry to an entry immediately after the selected entry if the hash value of the selected entry is greater than the hash key value;

decrementing the address corresponding to the hash key value if the hash value of the selected entry is greater than the hash key value; and

copying the data into an entry immediately after the selected entry if the hash value of the selected entry is greater than the hash key value.

59. The method of Claim 58, wherein the database comprises an Internet Protocol Security (IPSec) security association database and the plurality of selector values comprise IPSec selector fields.

60. The method of Claim 58, wherein the database has a size of about four times a maximum number of supported security associations, the most significant bits comprises the two most significant bits and evaluating most significant bits comprises determining if the two most significant bits of the hash value of the current entry are "11" and the two most significant bits of the hash key value are "00" or if the two most significant bits of the hash value of the selected entry are "00" and the two most significant bits of the hash key value are "11".

61. The method of Claim 58, further comprising:

comparing the selected entry to the data to determine if a duplicate entry is to be inserted into the database; and

returning a failure if a duplicate entry is to be inserted into the database.

62. The method of Claim 58, further comprising copying the data to the selected entry of the selected entry is empty.